NEW NEU III
足球机器人系统
用户手册

东 北 大 学

人工智能与机器人研究所

2002 年 2 月

目　录

一、足球机器人系统概况

1.1 微型机器人足球简介

微型机器人足球的赛场长 1.5 米，宽 1.3 米，比乒乓球台略小，场地画有中线、中圈和门区。每队由三个边长不超过 7.5 厘米的立方体形的遥控小车（机器人）组成。它们的任务就是将橘红色的高尔夫球（足球）撞入对方的球门而力保本方不失球或少失球。比赛规则与人类足球相似，也有点球、任意球和门球等。只是因电池容量有限，每半场为 5 分钟，中间休息 10 分钟。下半场结束时若为平局，则有 3 分钟的延长期，也实行突然死亡法和点球大战。与人类足球的明显不同之处在于球场四周有围墙壁，所以没有界外球，而在相持 10 秒后判争球。

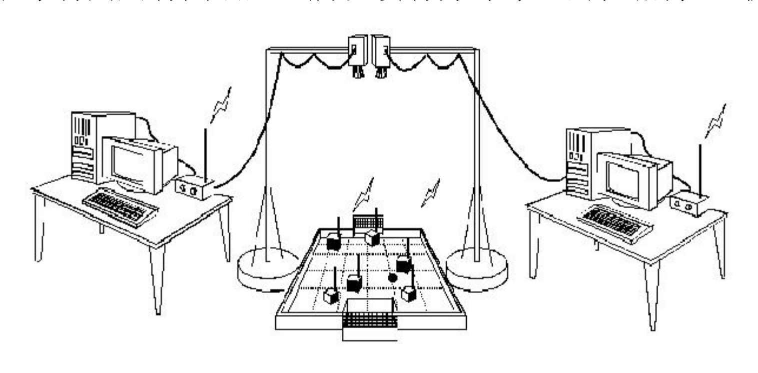


图 1.1 微型机器人足球系统总图

足球机器人系统，在硬设备方面包括机器人小车、摄像装置、计算机主机和无线发射装置（见图 1.1）。从功能上分，它包括机器人小车、视觉、决策和无线通讯 4 个子系统。

机器人小车由车架、车轮、电机、减速机、测速码盘、驱动电源、单片机控制电路与无线接收模块等构成。它可以按着主机发出的命令控制左、右轮转速，以保证按预定的轨迹运动。

视觉子系统是机器人的眼睛。它由悬挂在球场中圈上空 2 米的摄像头摄取图像，由装在主机内的抓图卡将图像数字化，送入主机内存，再由专用软件对图像进行理解。由于双方各有不同颜色的队标（黄色或蓝色），而机器人也有不同的队员色标,这样计算机就可以通过颜色分割辨识出全部机器人与球的坐标位置与

朝向。也就是进行模式识别。

装在主机中的决策子系统根据视觉系统给出的数据，应用专家系统技术，判断场上攻守态势，分配本方机器人攻守任务，决定各机器人的运动轨线，然后形成给各小车左右轮轮速的命令值。

无线通讯子系统通过智能通讯卡得到命令值，再由独立的发射装置与装在小车上的接收模块建立无线通讯联系，遥控场上各机器人的运动。

在机器人足球比赛过程中，上述 4 个子系统以每秒二、三十次，甚至更高的速率连续运行，人不得干预。因此，这完全是一场软硬件的较量，是一种高技术的对抗。

1999 年在巴西举行的第四届机器人足球世界杯赛上，东北大学的牛牛机器人足球队两胜巴西队，取得了微机器人足球赛的第 5 名和标准动作比赛冠军。

2000 年牛牛 I 型系统通过辽宁省科技成果鉴定。由国内人工智能和机器人领域的知名学者组成的专家组一致认定：该系统在总体性能上达到国际同类产品的先进水平，并填补国内空白。

1.2 微型足球机器人系统（III）基本配置

硬件配置（按一个队）见图 1.2

(1)PAL 制式彩色 CCD 摄像头一个

(2)镜头一个

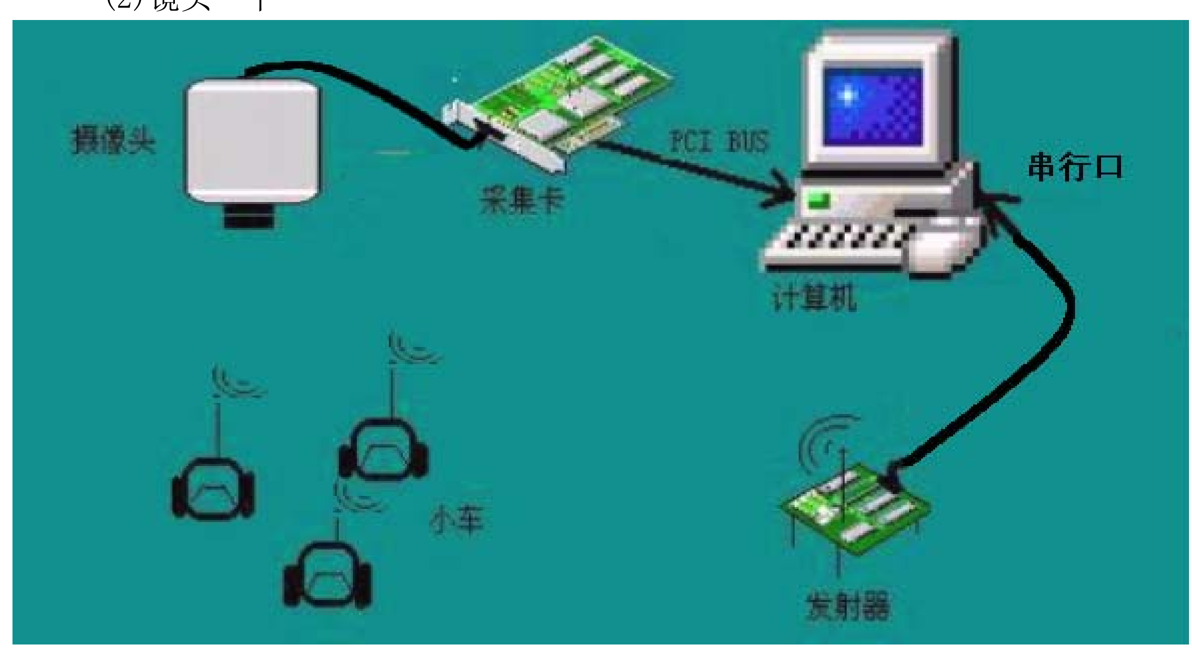


图 1.2 足球机器人系统的主要设备

(3)OK-C30 抓图卡一块

(4)机器人小车 III 型 4 只（其中 1 只备用）

(5)无线发射装置一套

(6)球、球台等辅助装置一套

(7)奔腾计算机   P3-350 (内存 ≥ 64M) 一套(自备)

软件配置：

(1)VC++6.0（自备）

(2)OK-C30 软件包

(3)NewNEU 3.0 软件

二、系统安装

2.1 硬件安装

- 关闭计算机电源，拔掉所有电源线；关闭视频输入设备电源。
- 打开计算机机箱。
- 将 OK-C30 图像采集卡插入任何一个空闲的 PCI 插槽。
- 关闭计算机机箱。
- 拿出随系统附送的视频输入线，D 型头一端与 OK-C30 的十五孔插座相连，BNC 插头一端与视频输入设备相连（黄色标记线为缺省标记线）。
- 打开视频输入设备电源开关。
- 打开计算机电源开关。

本图像采集卡可以接受 CVBS 视频信号和 S-Video 视频信号（Y/C 分离信号），最多可连接六路 CVBS 视频信号或三路 S-Video 视频信号。连接方式详见 OK-C30 用户手册。

2.2 软件安装

1) 安装图像采集卡驱动程序
2) 运行安装盘的 Setup.exe，安装采集卡管理及演示程序。

三、NewNeu3.0 软件简介及界面操作

3.1 主对话框界面



图 3.1 系统控制面板

控制面板如图 3.1 所示，各部分的功能如下所述：

1、系统菜单的使用说明

（1） 文件

其下拉菜单有：打开、新建、保存、另存为，功能有待补充

（2） 编辑

其下拉菜单有：撤销、剪切、复制、粘贴，功能有待补充

（3） 查看

其下拉菜单有：工具栏、状态栏

（4） 帮助

有关 NEWNEU 系统的版本信息

（5） 设置

进行视觉、策略、通讯、仿真程序的一些设置

2、各种模式的使用说明

（1） 比赛类型

有标准动作、微型、超微型，实现不同的比赛类型

（2） 参赛人数

根据实际情况来设置比赛人数

（3） 开球方

在比赛时，根据实际情况来决定哪方开球

（4） 我方半场

根据实际情况来决定我方处于左半场还是右半场

（5） 赛事选择

进行实际比赛还是仿真比赛的选择

（6） 开球模式

根据比赛规则来决定采用哪种开球方式

（7） 策略选择 A//无效

根据比赛对手及场上形势来判断采用什么样的策略

（8） 策略选择 B//无效

选择实际比赛是几对几比赛，即 1 对 1、3 对 3 还是 5 对 5 比赛

3、各按钮的使用说明

（1） 开始

进行初始化信息的设置**?**

（2） 停止

用于停止比赛过程

（3） 退出

退出比赛系统

（4） reset

重新设置决策的一些初始化信息

（5） thread start

比赛正式开始

（6） 应用

在面板上有车号、球与轮速，选择不同的车号及球标号，可以在车号和速度的标识的下方得到车的位置与轮速信息以及球的位置信息；在调试时，可以随时设置车和球的位置信息以及球速，按应用按钮即可使设置生效

（7） OK

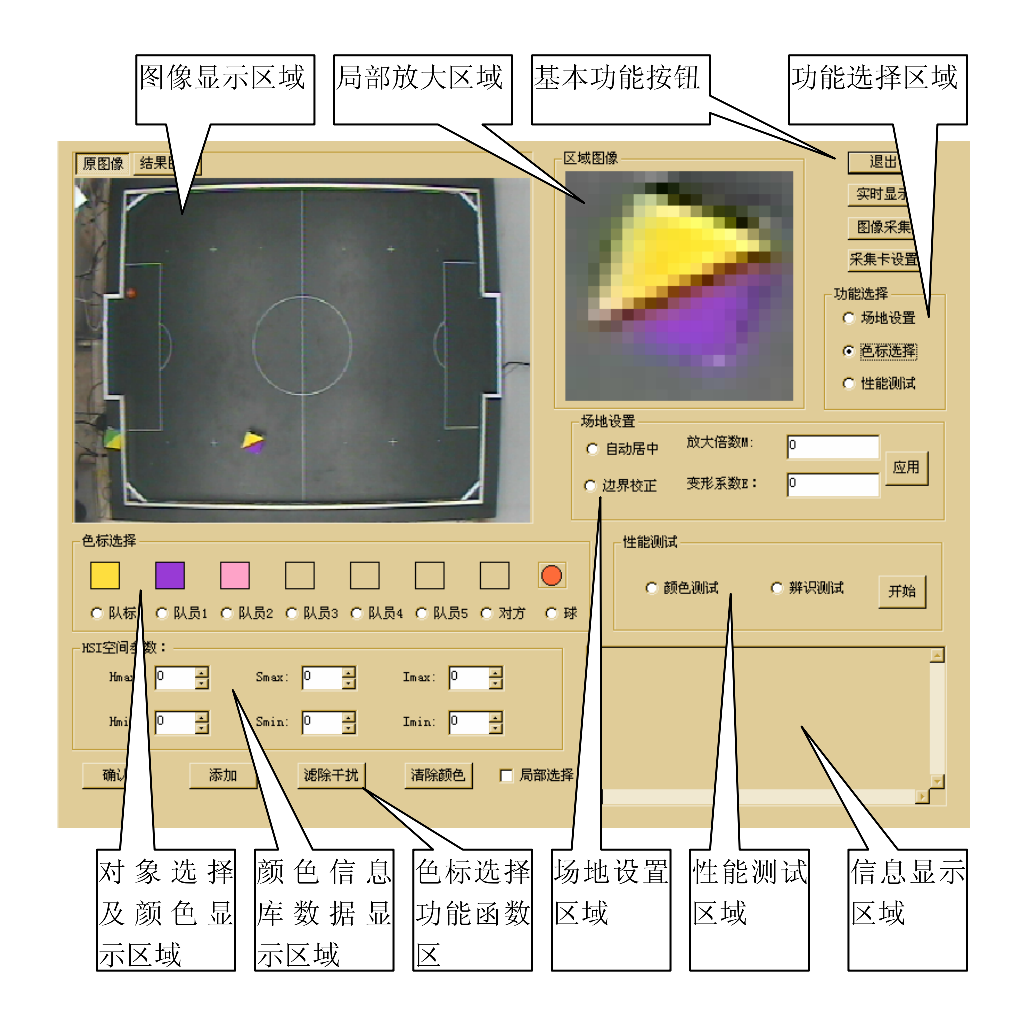
使开球方、开球模式、我方半场及赛事选择生效

3.2 各个子系统的使用说明

3.2.1 视觉子系统的使用说明

（注意：视觉子系统中黑体部分为特别需要注意的操作，请务必按说明进行操作）

NewNeuIII 视觉系统人机交互界面是视觉子系统用来通过人机交互的方式



完成初始化任务的界面。如上图所示：

（1） 基本功能按钮

✧ 退出：退出人机交互界面

✧ 实时显示：如果处于冻结状态则实时显示，如果处于实时显示状态则冻结显示。

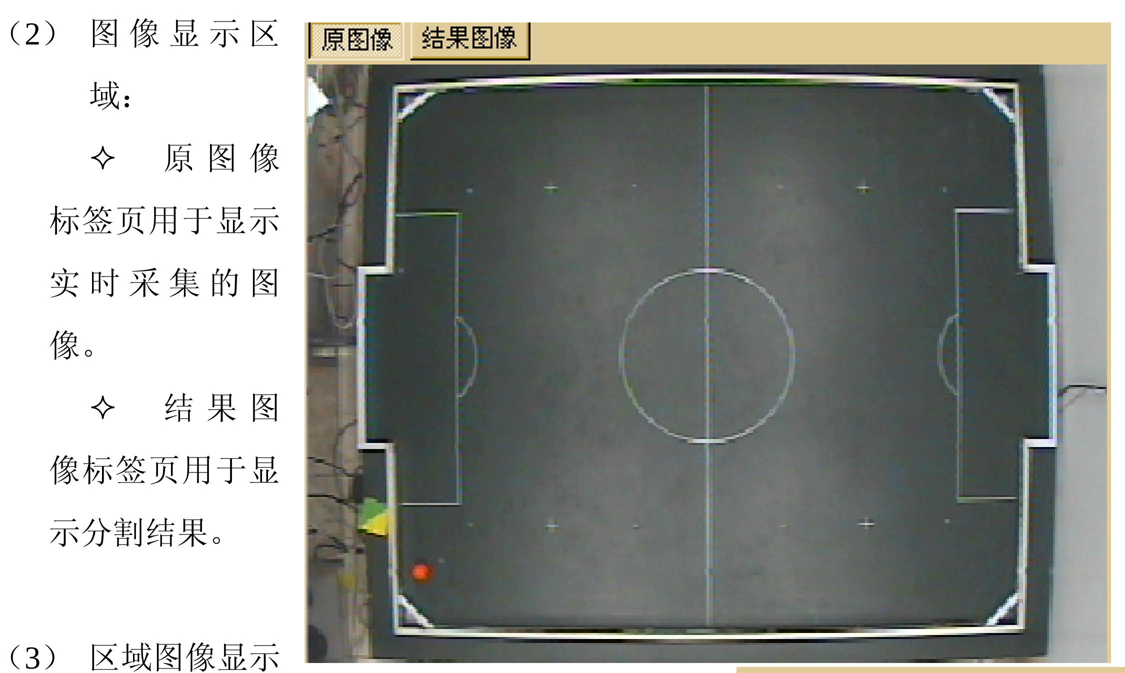
✧ 图像采集：采集并显示一幅图像。

✧ 采集卡设置：设置图像采集卡的参数。

（2） 图像显示区域：

✧ 原图像标签页用于显示实时采集的图像。

✧ 结果图像标签页用于显示分割结果。

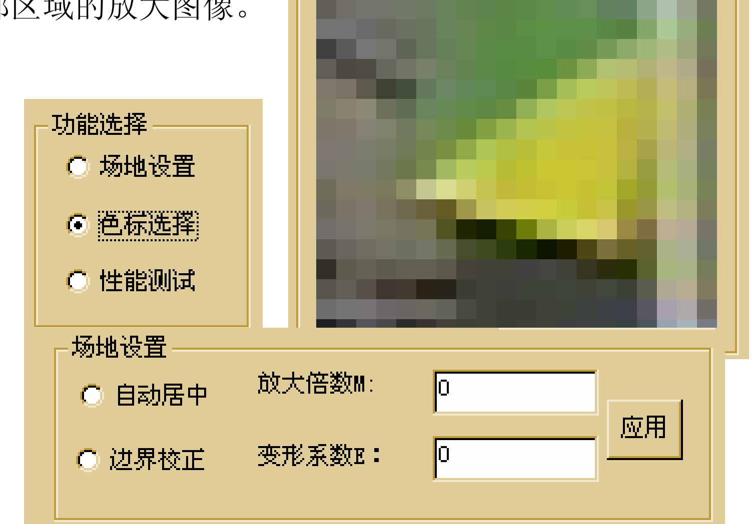


（3） 区域图像显示区

在色标选择时用于显示局部区域的放大图像。

（4） 功能选择区

根据单选按钮的选项分别激活场地设置、色标选择和性能测试功能。

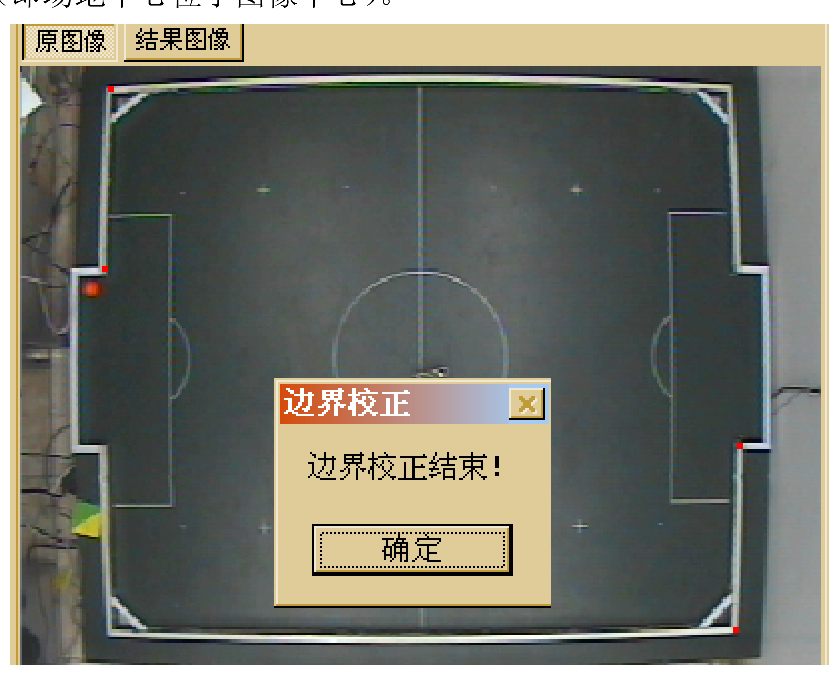


（5） 场地设置区

当功能选择**选中场地设置**后，激活场地设置功能。场地设

置共有两项：

自动居中：选中自动居中后，在**原图像中**用鼠标左键单击场地中心，可实现图像的自动居中（即场地中心位于图像中心）。



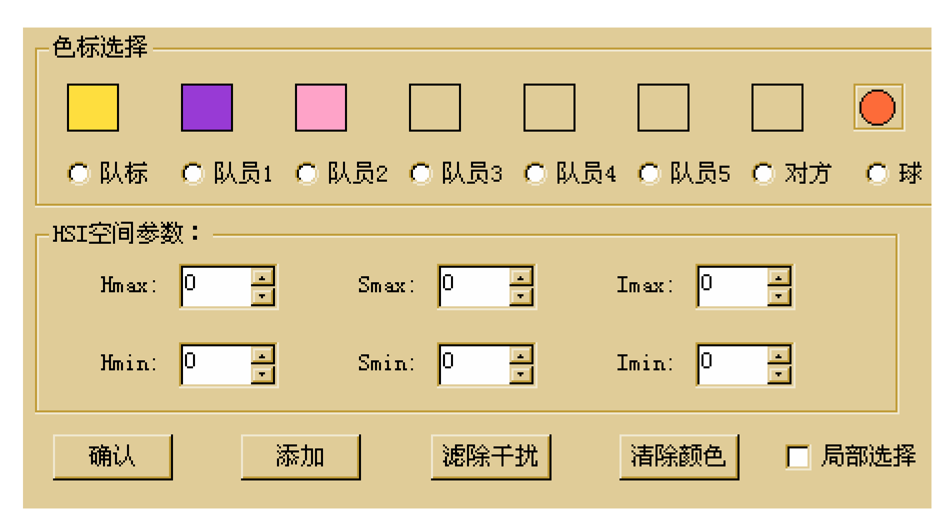
边界校正：选中边界校正后，在场地设置区域中我们可以看到当前的放大倍数 M 和变形系数 E。如果想要重新进行场地校正，则可以在原图像中**沿逆时针方向用鼠标左键选中场地的左上角、门区与场地交界的左上角，场地右下角、门区与场地交界的右上角**。系统会自动将选中的点以红色来标记。如上图所示。选中四点后，单击应用按钮程序开始计算场地的变形参数，并进行校正。结束后会显示边界校正结束的提示。单击确定以完成校正。选择结果图像标签可以看到校正结果。如果对校正结果不满意，可以重复上述操作，还可以通过直接修改 M 和 E 的值得方法来调整，参数修改后，单击确定以完成校正。选择结果图像标签可以看到校正结果。



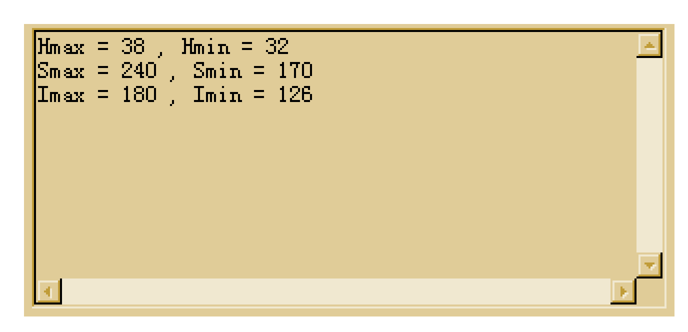
（6） 色标选择区

色标选择区包括对象选择及颜色显示区、颜色信息库数据显示区和色标选择功能函数区。当功能选择选中色标选择后，激活色标选择功能。本系统支持对我方队标、5 个队员标志、对方队标和球建立颜色信息库。建立颜色信息库的过程如下：





首先选中一个对象，在颜色信息库数据显示区就会显示该对象当前的颜色信息库信息。在原图像中单击鼠标左键选中该对象的色标中心，在区域图像显示区中就会显示该色标的放大图。将鼠标移至该区域后，鼠标光标变为套圈。用套圈的中心套中色标的中心，点击确认按钮。系统自动对采样区域的颜色信息进行分析，并在信息显示区域给出分析结果。此时选中结果标签页，可以看到根据当前采样颜色信息对整幅图像进行分割的结果。如果对结果满意，单击添加按钮即可将当前信息加入到所选特征对象的颜色信息库。因为特征对象的颜色信息库是在 HSI 空间的连续分布的，所以我们只需要在场地中几个有代表性的区域进行几次采样即可建立比较完整的颜色信息库。另外，还可以通过直接修改颜色信息库数据的方式来调整颜色信息库。一般情况下我们可以适当放大 H、S、I 的范围，点击确认来完成修改。



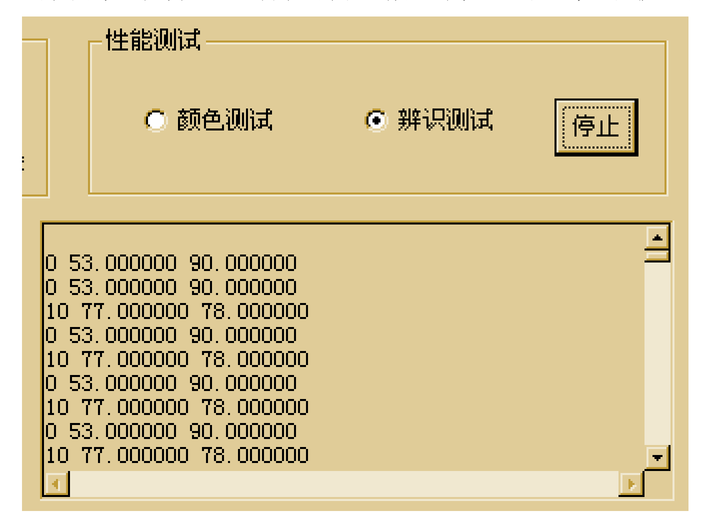
我们可以单击清除来清除所选特征对象的颜色信息库，目前，滤除干扰的功能还未完善，用户需要通过直接修改颜色信息库数据的方式去除干扰。具体请见性能测试区的颜色测试。



（7） 性能测试区

当功能选择选中性能测试后，激活性能测试功能。性能测试包括两项：

颜色测试：颜色测试可以测试所选特征对象的颜色信息库的完整程度，将待测特征对象的色标置于场地内任意位置，选中结果图像标签页，在该区域内单击鼠标左键，系统自动采集一幅图像，并根据待测特征对象的颜色信息库对图像进行分割，并显示分割结果。如果出现空洞，或者辨识结果不够丰满，可以选择色标选择，在该区域对该色标进行再次采样，以补充颜色信息库。当出现干扰的时候，我们可以选中色标选择，然后选中结果标签页。鼠标左键单击干扰区域，在局部图像区域中显示干扰的放大图像，一般情况下，干扰区域较小，我们需要选中局部选择选择框，以对选中的局部区域进行分析，分析结果显示在信息显示区域，用户根据分析结果，调整特征对象的颜色信息库的范围，以去掉干扰。



辨识测试：用来测试系统的实时辨识性能，选中辨识测试，单击开始按钮，开始实时辨识，此时该按钮变为停止按钮。系统连续采集图像并进行辨识，在图像中，以蓝点显示辨识出的队标、球、对方队标的中心，用红点显示辨识出的队员标志的中心，并在信息显示区域显示辨识的数据结果。单击停止按钮，停止实时辨识，此时系统自动计算并显示辨识周期。



3.2.2 小车与通讯使用说明

3.2.2.1 Mirosot 机器人车体（III）型

1、 性能指标

(1)电机特性（微型直流电机）

电机工作电压 7.2V

电机额定转速 6000rpm

电机额定电流 120mA

(2)电池特性

电池容量 680ma/h

充电电流 500mA

充电时间 约 1.4 小时

电池一般能够维持车体正常工作 30 分钟

(3)车体特性

重量 0.4 千克

轮直径 4.5cm

轮的最大线速度 120cm/s

检测精度 1 个脉冲相当于 0.0614cm

2、车体的结构与功能要求

车体的机械结构及传动齿轮对整个机器人的灵活性起着不可忽视的作用。为了获取较好的机动性和灵活性，本系统采用双电机分别驱动左右两轮的方式。除了分布在车体左右两侧的主动轮外，在车体的前后端各有一个支撑轮以保持行进当中车体的平衡。这样的机械结构布局使机器人小车很容易实现以自身为圆心的旋转运动。机器人小车的重心应尽可能的低，对称性要好，这样才能够保证运动的平稳性及减小被撞翻的可能。车体采用铝金属框架以增大其坚固性。电机的选型可选择小型直流电机。小型直流电机的规格较多且很便宜。

根据比赛规则的要求，MiroSot机器人小车的尺寸应不大于 7.5×7.5×7.5cm^3 。为了满足足球机器人比赛的要求，机器人小车的运动性能显得尤为重要。机器人小车应具备高度的机动性和灵活性，能够快速实现前进、后退、转角、停车等基本动作。由于比赛环境比较恶劣，存在各方面的干扰，如电机的电火花对无线接收机部分及车载CPU的干扰，强烈碰撞造成的电缆插接件及机械零部件的松动。所有这些不仅对机器人的结构，而且对控制系统的设计都提出了严格的

要求。

概括地说，机器人小车应能准确地接收上位机指令，并根据指令要求迅速完成决策子系统的意图（带球、射门、拦截等技术动作），为此机器人应具有以下两大功能：

- 指令的接收
- 调速及转角控制功能

3、 速度控制系统

(1)控制命令的格式

如果把整个足球机器人系统看作一个闭环控制系统，则小车在这个闭环系统中充当执行机构的角色。决策系统根据场上的形势作出决策，然后分派给每个小车来执行这一决策意图。主机每约 40ms 发送一次控制命令。主机发送给每个小车的控制命令字中包括三个字节，第一字节给出的是小车的标识，第二字节给出的是左轮速度，第三字节给出的是右轮速度。

这样三个命令字作为主机和小车通讯数据格式。主机发命令时，小车接收到

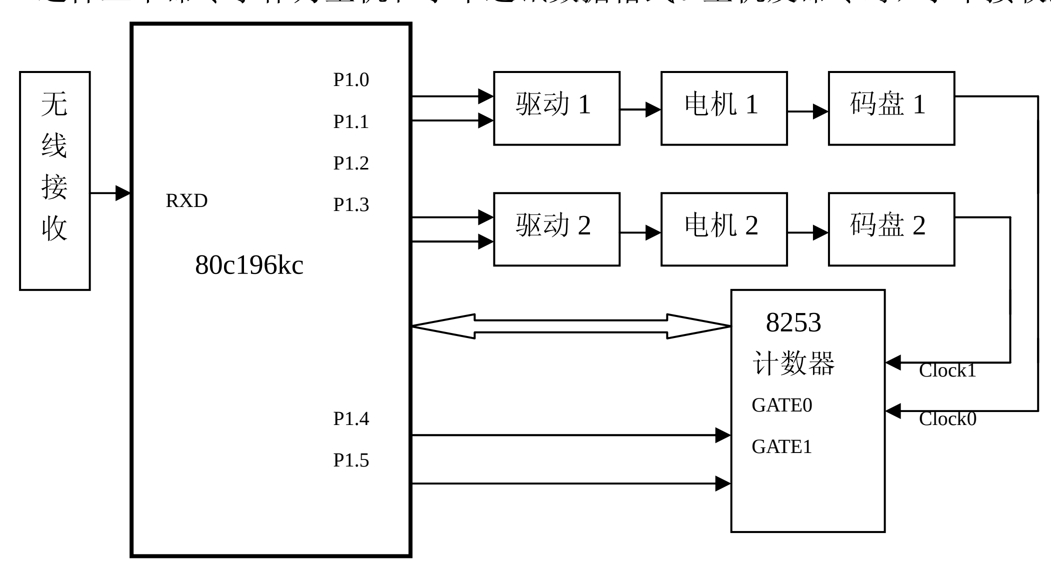


图 3.2 控制单元结构图

标识后与本机地址码相比较，若相等则接收下面两个字节。实际上这是一种广播式的通讯方式。第二和第三个字节分别为小车的左右轮速度给定。次高位 D6 给定速度的方向，D6 =1 反转，D6=0 则正转。低 6 位 D5~D0 为小车给定的绝对转速。实际上表示在一个控制周期中速度给定值，速度范围在 0 到 60 之间。

(2)控制单元

机器人控制系统以 80c196kc 单片机为核心，包括驱动，速度检测，无线接收，看门狗等环节构成，其基本控制框图如图 3.2。

在本系统中车载 CPU 主要完成以下几种功能：

- 经无线接收机接受上位机的指令；
- 根据速度给定和实测值按预定控制算法得到控制量，并实现 PWM 方式调速，完成车体本身的闭环控制；
- 复位 WATCHDOG 计数器。

(3)闭环控制系统的结构图

小车本身作为一个独立的运动物体，它主要完成对命令的接收，译码和速度的闭环控制。其控制系统的结构如图 3.3 所示：

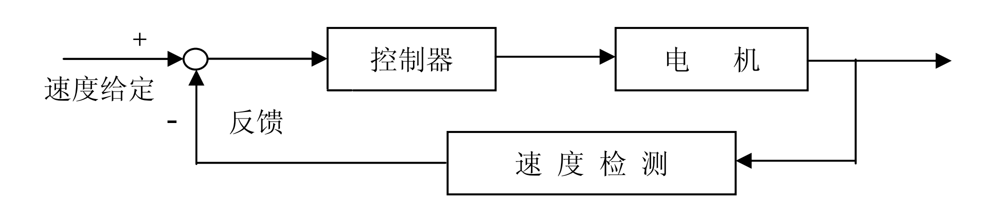


图 3.3 单片机速度控制系统结构图

(4)电机驱动环节

采用集成电机驱动芯片 L298 来驱动电机。L298 为双桥高电压大电流功率集成电路，可以驱动继电器线圈、直流电机、步进电机等感性负载。

(5)速度检测环节

光电编码器可用来测量一个旋转设备的角位移。它的输出是脉冲信号，因此可用简单的办法转换成数字量，向微控制器提供位置和速度反馈量以实现车体运动的控制。

由光电开关得到的脉冲经过整形的送 8253 的 CLK 端，利用 8253 对脉冲进行计数，其中 GATE0，GATE1 可以控制 8253 的计数的启停，这样可以方便地读取一个控制周期的脉冲数（反馈速度）。

4、 电源与充电装置

在赛场上，每个机器人都是一个相对独立的运动物体，这就要求它们都应具有各自的电源装置。电源存储的能量应足以保证机器人小车参加比赛所需能量。由于尺寸的限制，机器人小车采用同一电源为电机和控制电路供电，电源由六块额定电压为 1.2 V 的镍氢电池串联组成。由于电池的电压会随着电路和电机的消耗而变化，所以必须经稳压后才能为电路板供电。

由于电源的特殊性，市场上很难买到相匹配的充电装置。可根据镍氢电池的充放电特性，及所选择电池的充电电流、容量自行设计充电装置。本充电装置是直接对电池进行恒流充电。

5、 应用指南

(1)车号的设定

车号可通过车体上层板的拨码开关来设定，具体设定见下表：

车　　号	00	01	02	03	04
拨　码　1	OFF	ON	OFF	ON	OFF
拨　码　2	OFF	OFF	ON	ON	OFF
拨　码　3	OFF	OFF	OFF	OFF	ON

注意：改变车号后应重新对小车复位。

(2)电池充电

充电装置具有六个充电接头，充电器的输入电源接到18V、100瓦的直流恒压源上，可同时对6路电池进行充电。当电池插入充电器的插孔后，按下复位键，稳压芯片接通电源，输出500毫安的恒流，给电池充电，绿色指示灯亮，当充电时间到，继电器断开，停止对电池充电，红色指示灯亮。要想再次对电池充电，则重新按下复位键。

(3)通讯模块

为了运输的方便，通讯模块和天线是可以插拔的，插入时应注意方向，通讯模块插针较少的一面应靠近天线一方。

注意：小车上电前一定要先开发射器，不然会出现小车失控的现象。

6、 问题及处理

(1)当发现车速明显下降时，请及时充电，正常工作中电池电压应在 7 伏以上（带载）。

(2 如发现车体显示灯不亮，则检查电源插头和电源开关是否存在问题

(3)当发现一个小车接不到命令时，检查天线是否松动。如果所有小车都接不到命令，则检查发射器的天线是否接触良好，通讯电缆是否和通讯卡插接良好，且通讯电缆是否正常导通。

(4)车轮转动不灵活，可检查减速器内是否有赃物，若赃物太多可用汽油清洗。

3.2.2.2 无线通讯子系统

1、无线发射器

为了充分利用主机的现有资源，减少不必要的开销，利用主机的串行口与无线发射器相连，就可以把决策系统的指令信息发送给机器人。其原理图如图 3.4

所示。

在无线发射器中，可通过跳线开关 S2 来选择发射频率。在发射时，决策系统根据视觉所提供的辨识信息做出策略，然后形成控制命令，通过主机的串行口

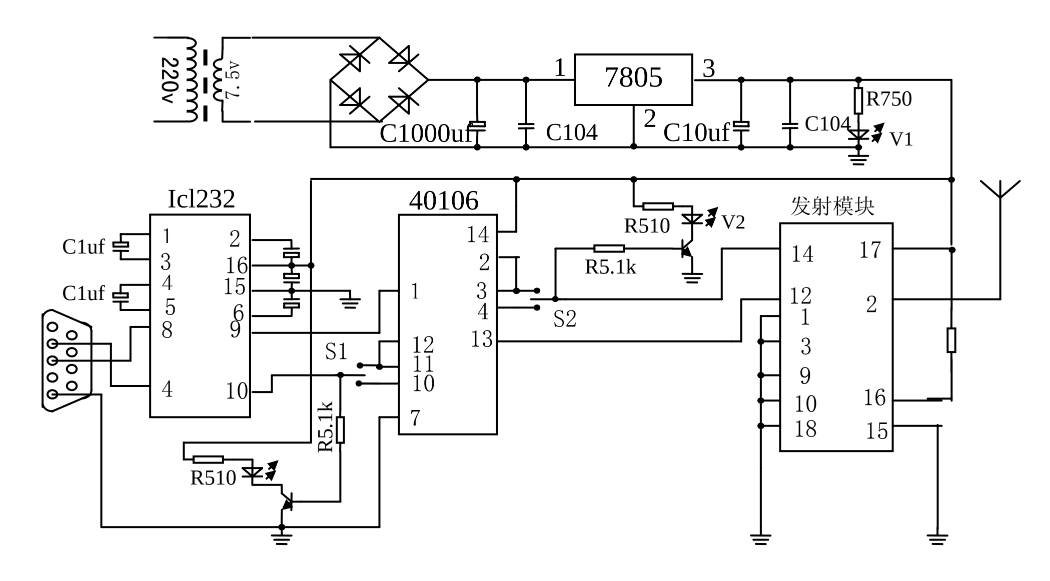


图 3.4　　无线发射器原理图

给无线发射器的电平转换芯片 ICL232，将 232 电平变为 TTL 电平，然后调制发射模块，以无线电波的形式发射出去。在此发射器中，可通过指示灯 V1 的点亮来显示电源工作是否正常，通过指示灯 V2 的闪灭来显示数据发送是否正常。

2、　无线接收器

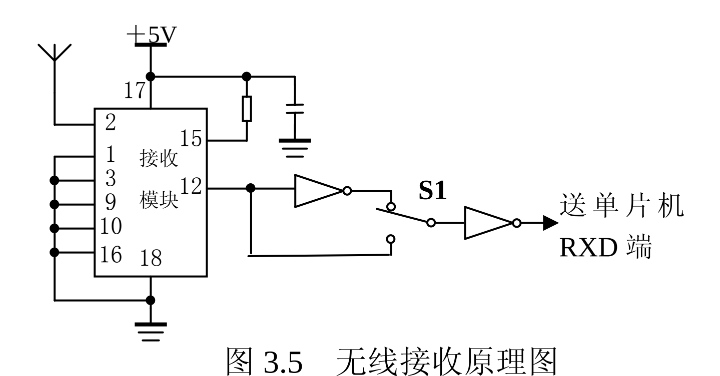


图 3.5　无线接收原理图

图 3.5 给出了无线接收器原理图。将 15 脚发射允许端接为高电平（禁止发射功能）和 16 脚接收允许端接为低电平（使能接收功能）。接收模块 12 脚 RXD 端为接收到的数据信息，经整形后直接送单片机串行输入端口。此无线接收器可通过跳线开关 S1 来选择接收频率。

3.2.3 决策子系统开发

1、开发环境与说明

NewNEU3.0 系统是在 Windows98 操作系统上，使用 VC++6.0 开发的。这就要求决策系统的开发环境是 VC++6.0。

在比赛开始前双方要选择场地，这里定义为左半场（LeftArea）和右半场(RightArea)，比赛场地通过摄像头显示在屏幕上，参照计算机屏幕定义左右半场。通过坐标变换总可以认为左半场为我方半场（即后场），右半场为对方半场（即前场）。选择一种策略，然后按“GAME START”键开始。在比赛中止时，可根据比赛情况，更换策略选择。具体操作见人机界面部分。

为方便描述策略，定义我方半场为后场，对方半场为前场，以场地左下角为坐标原点，其坐标系如图 3.6 所示。

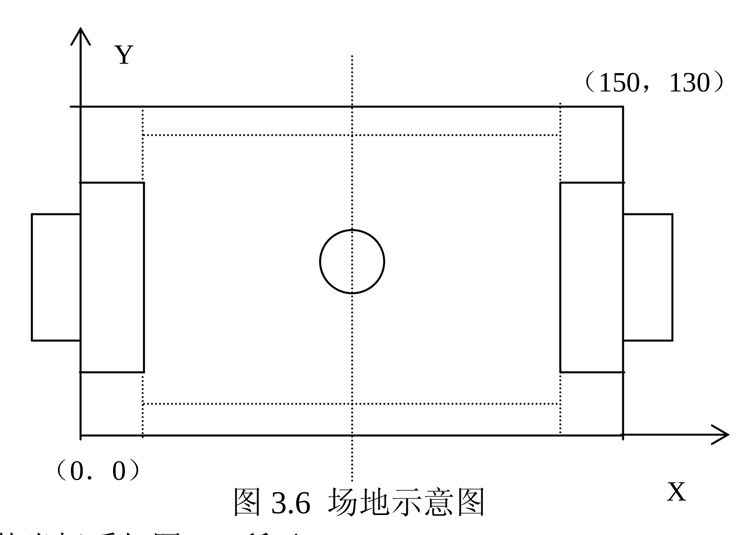


图 3.6 场地示意图

2、 NewNEU3.0 决策子系统结构

决策系统所能得到的信息仅是由视觉系统传递来的球的位置以及球员的位置和方向信息，而决策系统传输给小车的信息仅为小车的左右轮速，面对这样一个问题，我们利用面向对象的程序设计思想把整个决策系统看成一个复杂的对象，然后进行分解、细化，最后使每一部分简单化，最终形成如图 3.6 所示的决策类层次。

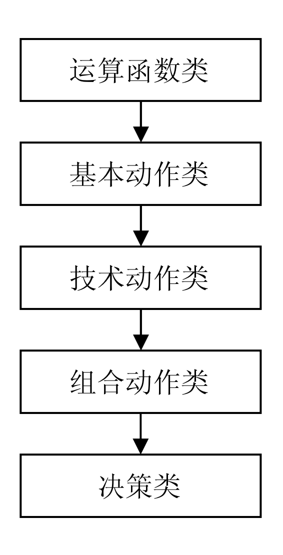


图 3.7 整个决策类层

图 3.6 中所示的类从顶到底（即由基层到高层）依次继承，高层可以继承基层，但基层不能继承高层，高层类中方法的实现需要基层类中方法的支持，其中运算函数类的属性是整个决策所用变量信息的一个结构体，如小车、球所在区域变量，角色变换标识变量等等，方法主要是一些解决类似于几何运算和反应场上状态信息的函数，如求球所在的区域号码，预测球的轨迹、速度和方向（ForcastBall）等等；基本动作函数类的方法完成如原地转动（Turn）、转到定角(TurnToAnglePD)、转到定点(TurnToPointPD)、到定点（ToPositionPD)、到达定点有一定的速度函数（Move2Pt）等等，其属性是可调参数的结构体；技术动作函数类中封装一些比较高级的动作，如完成射门（Shoot）、防守(Defence)、边界处理(BoundProcess)等功能；组合动作函数类是更高层次的类，其方法用来完成多车协作动作，如点球大战（LefPenaltyOnly）、争球（FreeBallKick）等动作；决策类是整个决策系统的最高层，是整个决策的核心部分，他的方法就是用这些底层类来实现决策者的意图，如信息预处理、态势分析、角色分配、动作实现等。

3、 函数类的属性及方法

（1）COperationFun—运算函数类

a、属性：

```
typedef struct _DecisionParamter
{
  int        sign4;//辅车没被阻挡标志
  int        numPara;//保留球和车的信息的周期数
  int        nBound_Area;//球所在的区域号
  int        nMRBound_Area,nARBound_Area;//主车、辅车所在的区域号
  int        nMF,nAF;//主车号、辅车号
  int        charSelectSign,passBallSign,passBallSignOver;//角色选择
                                   //标志、传球标志、传球结束标志
  int        nMFAction;//主车的动作
  int        charChange,fieldChange;角色转换和场地转换标志
  int        StartSW;开始
```

```
int         nPositionBall;
char        Command[3][3];//setCommand 用来写待发送的数据
double      dist[2];//主车、辅车与球的距离
BOOL        nReset;//是否归位,FALSE
dbPOINT     MRtarget,ARtarget;主车、辅车的目标点
BallInformation      ball;球的信息
BallInformation      oldball;
BallInformation      proball;下 step 个周期的球的位置
BallInformation      oldBallPt[7];前 7 个周期的球的位置信息
dbPOINT          MRobotInfo[7];前 7 个周期的主车的位置信息
dbPOINT          ARobotInfo[7];前 7 个周期的辅车的位置信息
RobotInford      Robot[7];保存视觉传来的数据
FORCASTBALL      pBall;
```

}**DecisionParamter;**

b、方法：

计算直线和球门的交点

double GetCrossPtWithGLLine( );

计算球所在的区域号码

int MarkBoundArea(BallInformation ball);

预测球的位置

BallInformation ForcastBall(BallInformation ball, int step);

计算中分点

dbPOINT GetMidPoint(dbROBOTPOSTURE * pRobotInford,
BallInformation ball, double deta_y);

dbPOINT GetMidPoint(dbROBOTPOSTURE* pRobotInford, BallInformation
ball,dbPOINT shoot_target,double deta_y);

预测球的运动直线、方向和速度

int forcastball(dbPOINT *pPoint, int nPosition, int nCount,
FORCASTBALL *pBall);

（2）CBasicAction: public COperationFun—基本动作函数类

a、属性：

typedef struct _AngleParameter

{

double Kp; //用 PD 法调节转动时的比例值

double Kd;// 用 PD 法调节转动时的微分值

double AngleError;//视觉所能分辨的角度，现为 5 度

double MaxAngleSpeed;//小车的最大旋转边缘线速度(cm/s)

double MaxMoveSpeed;//小车的最大移动线速度（cm/s）

}AngleParameter;

typedef struct _MoveParameter

{

double max_distance;//在车和目标点距离<=max_distance 时减速

double max_distanceG;//同上（守门员用）

double V_MAX;//最大速度

double V_max;//比赛时的正常速度

double max_angle; //在车和目标方向角的夹角<=max_angle 时减速

double kp4pospd;//ToPositionPD 函数的 kp 参数

double kd4pospd; //ToPositionPD 函数的 kd 参数

double kp4pospdG; //同上（守门员用）

double kd4pospdG; //同上（守门员用）

}**MoveParameter,MOVEPARAMETER;**

b、方法：

定点旋转

int Turn(double dbVelocity, int clock , dbLRWheelVelocity *pSpeed)；

使小车按指定时针方向转到所要求的角度

int TurnToAnglePD(dbROBOTPOSTURE *pRobot, double dbAngle,int clock, dbLRWheelVelocity *pSpeed)；

具有 PID 校正的按指定时针方向转向定点

int TurnToPointPD(dbROBOTPOSTURE *pRobot, dbPOINT Point, int clock, dbLRWheelVelocity *pSpeed);

小车沿某一方向角移动

int MoveOnAngle(dbROBOTPOSTURE *pRobot , double Angle, dbLRWheelVelocity *pSpeed)；

以余弦曲线到定点

int Move2Pt(dbROBOTPOSTURE * pROBOTPOSTURE, dbPOINT Target, dbLRWheelVelocity * pWheelVelocity);

以余弦曲线按给定的速度到定点

int Move2Pt(dbROBOTPOSTURE * pROBOTPOSTURE, dbPOINT Target, double speed, dbLRWheelVelocity * pLRWheelVelocity);

具有 PD 校正的走弧线到定点

int ToPositionPD(dbROBOTPOSTURE * pROBOTPOSTURE, dbPOINT Target, double vBase, dbLRWheelVelocity * pLRWheelVelocity);

具有 PD 校正的走弧线到定点

int ToPositionPDGoal(dbROBOTPOSTURE* pROBOTPOSTURE, dbPOINT Target, double vBase, dbLRWheelVelocity* pLRWheelVelocity);

送命令到命令数组

void SetCommand(dbLRWheelVelocity * pLRWheelVelocity, int RbNo);

（3）CSkilledAction : public CBasicAction—技术动作函数类方法：

在追球过程中，防止碰撞边界

int AvoidBound(dbROBOTPOSTURE *pRobot, dbLRWheelVelocity *pSpeed);

边界球处理

int BoundProcess(dbROBOTPOSTURE *pRobot, BallInformation Ball, int field, dbLRWheelVelocity *pSpeed);

小车回防

int Defence(dbROBOTPOSTURE *pRobot, ballInformation ball , dbLRWheelVelocity *pSpeed);

守门

int GoalieAction(dbROBOTPOSTURE * pRobotInford, BallInformation ball, dbPOINT proball, dbLRWheelVelocity *pSpeed);

射门

int Vect_MidShoot(dbROBOTPOSTURE * pRobotInford, BallInformation &ball, dbLRWheelVelocity *pSpeed);

传球

int Vect_MidShoot(dbROBOTPOSTURE * pRobotInford, BallInformation &ball, dbPOINT shoot_target, dbLRWheelVelocity *pSpeed);

避障到定点

int AvoidPt2Pt(dbROBOTPOSTURE* pRobotInford, dbPOINT &obsPt, dbPOINT &target,dbLRWheelVelocity *pSpeed);

（4） CComAction : public CSkilledAction—组合动作函数类方法：

归位

void RobotReturn2Pt();

单罚点球

void LefPenaltyOnly();

争球

void FreeBallKick();

罚点球或者任意球

void PenaltyOrFree_Kick(int nFlag);

门球

void LefGoalKick();

（5） Decisiontaking : public CcomAction—决策函数类方法：

初始化函数

void Initialize();

决策入口

void Start(RobotInford dmRobot[7],DEGame dmDEG);

MIROSOT 比赛

void MiroSot_DecisionMaking();

3 对 3 比赛中线开球

void　LefNormalDecision(int nFlag);

信息预处理

void　preProcess();

角色分配

void　charSelect();

动作选择

void　actSelect();

动作执行

void　actProcess();}

4、主要函数的算法描述

（1）基本动作类中一些函数实现

足球机器人的基本动作函数类似足球运动员所具备的基本素质，如跑动、站位、转身等。这里所涉及的基本动作主要包括：

✧ ToPositionPD　　（站位/到定点）

✧ TurntToAngle　　（转到定角）

✧ MoveOnAngle　（跑动/移动）

用户可以直接使用这些函数，其出入口参数的结构均在文件 Decisiontaking.h 中定义。也可自行开发新的基本动作函数或扩充基本动作函数，只需将函数声明和函数体分别放在文件 Subfun4win.h 中 Subfun4win.cpp 即可。

下面给出在算法描述中使用的一些变量：

robot:　机器人结构

ball:　球的结构

robotV:　机器人速度结构（Vl、Vr）

target:　目标点结构

dist_E:　距离偏差

theta_E:　角度偏差

Vl、Vr:　机器人左、右轮速度

x,y:　图 3.6 所定义坐标系中的 X,Y 坐标

Kp:　比例系数

Kd:　微分系数

Δtheta_E:　角度偏差的变化率

Vmax:　最大速度

Vbase:　同向速度

如图 3.8，主要算法如下所述：

1. **ToPositionPD**

Process：　控制机器人到定点

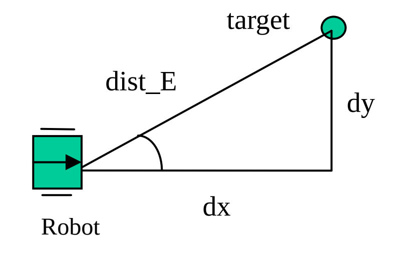


图3.8

Arithmetic：

1）同向算法：

```
If (dist_E>20)
        Vbase=Vmax;
Else
        Vbase=dist_E/20*Vmax;
Vdiff=theta_E/90*Vmax;
Vbase= Vbase*Vbase/(Vbase+ Vdiff);
Vl=Vbase-Vdiff;
Vr=Vbase;
```

2）PD 算法：

```
If(dist_E>20)
        Vbase=Vmax;
Else
        Vbase=dist_E/20*Vmax;
Vdiff= (Kp*theta_E+Kd*Δtheta_E);
Vl=Vbase-Vdiff;
Vr=Vbase+Vdiff;
```

2. TurnToAngle

Process: 控制机器人转向给定的角度

Arithmetic：

```
theta_E=atan2(dy, dx);
Vl= (Kp*theta_E+Kd*Δtheta_E);
Vr= (Kp*theta_E+Kd*Δtheta_E);
```

3. MoveOnAngle

Process: 控制机器人朝着给定的角度运动

Arithmetic:

```
theta_E= atan2(dy, dx);
```

```
If (theta_E∈[0,90] )
        Vl=Vmax*cos(theta_E);
        Vr= Vmax;
  Else If (theta_E∈[90,180] )
        Vl= -Vmax*cos(theta_E);
        Vr= - Vmax;
  Else If (theta_E∈[180,270] )
        Vl= -Vmax;
         Vr= - Vmax*cos(theta_E);
  Else
        Vl= Vmax;
         Vr= Vmax*cos(theta_E);
```

（2）技术动作类中一些函数实现

在基本动作类基础上开发技术动作类。技术动作主要包括：

- ✧ Vect_MidShoot　　（射门动作）；
- ✧ Defense　　　　　（防守动作）；
- ✧ Goalkeeper　　　 （守门动作）；

射门和防守动作有多种方法可以实现，如：状态转换、矢量域控制、中分线等方法。这里只是示例性的给出其中的一种方法（中分线法）。用户可参考此方法继续开发新的算法。（声明和原文件分别放置在文件 Decisiontaking.h 和 Decisiontaking.cpp 中。）

1．<u>Vect_MidShoot</u>

Process:　　利用中分线法进行射门

如图 3.9 所示，点 G 为球进入球门点，小车 A 与球 B 位于图中所示位置，小车 A 与球 B 中心连线段的中垂线与射线 $\overrightarrow{GB}$ 的交点为 C，C 点即为此时小车的瞬间目标点，让小车以余弦函数法向目标 C 点运动。从理论上

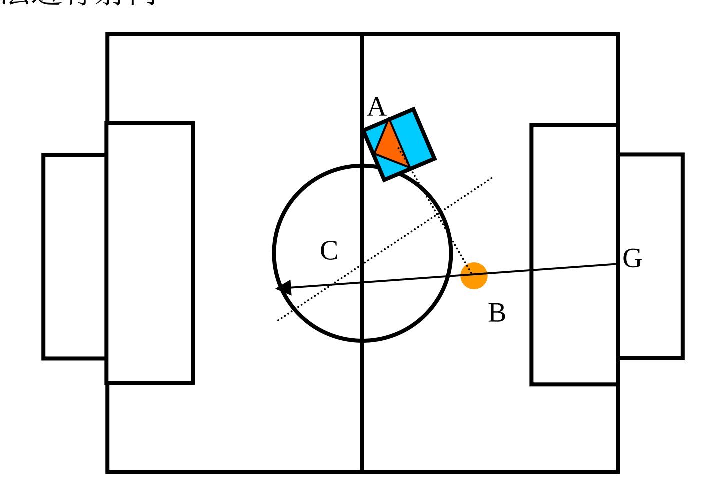


图 3.9 中分线射门法

讲，当小车中心与球心能无限制接近时，小车踢球时，瞬间踢球方向为射线 $\overrightarrow{BG}$ 。

上述过程很容易由计算机实现，其实现过程如下：

a.由已知视觉信息，求出小车 A 与球 B 中心的连线段；

b.求出线段 AB 中垂线与射线 $\overrightarrow{GB}$ 的交点 C；

c.用余弦函数法计算出小车 A 的左右轮速，驱动小车前进一个视觉周期；

d.判断小车是否能踢到球，是，踢球，否，重复上述过程

2. <u>Defence</u>

Process: 按照状态空间转换法控制机器人进行防守，如图 3.10 所示。

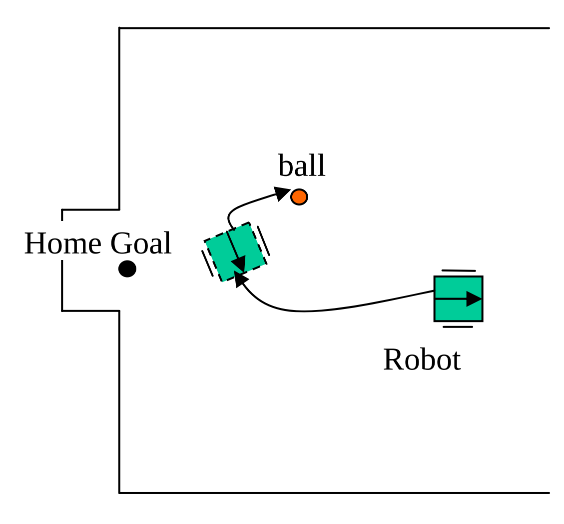


图 3.10

Arithmetic:

switch(sw)

{

case1: 如果 ball.x < robot.x，则机器人避开球向球的左侧移动，否则转到状态 2（sw=2）；

case2: 如果 ball.x > robot.x，则机器人向球左侧一点运动，当机器人与球较近时 sw=3；如果 ball.x < robot.x，则转到状态 1。

case3: 如果 ball.x < robot.x，则 sw=1，否则去带球，如果成功则 sw=4。

case4: 将球带向对方半场.

}

3. GoalieAction

Process: 按状态转换法控制机器人守门，如图 3.11 所示。

Arithmetic:

switch(sw)

{

case1: 球在我方半场，如果球在禁区（goalarea）则守门员向球跑去，以将球踢出门区，如果球在危险区（dangerarea）则守门员跑向球与goal 连线和 homeline的交点处。否则守门员在门区内跟踪球。球在对方半场，则守门员回到 homePt，sw=2。

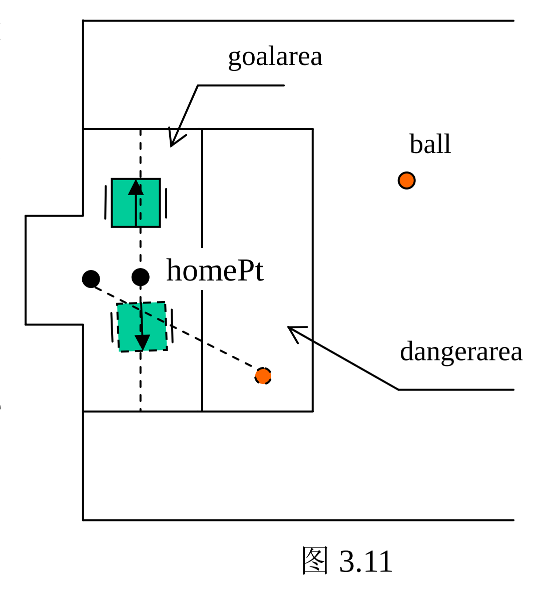


图 3.11

case2: 调整守门员的角度，防止守门员碰壁，如果球跑向我方半场则sw=1。

}

5、策略软件开发平台

在 NewNEU3.0 系统中策略软件给出了原代码，提供了一个开放性的平台。策略文件主要有：

- ✧ **Decisiontaking.h** ：策略函数头文件；
- ✧ **Decisiontaking.cpp**：策略函数原文件。

用户可以直接使用软件中提供的类成员函数，也可在所提供的各个类中添加不同的方法，从而可以进一步完善决策系统，开发自己的策略。

四、系统调试要点

进行系统调试就像训练运动员一样训练机器人。这个训练过程也就是算法改进、参数调整过程。

1. 遵循先局部后整体的原则。先调试基本动作类成员函数，再调试技术动作类成员函数，最后测试策略类成员函数。
2. 掌握 PID 调节原理，根据运行状况对参数进行调整。
3. 先粗后细，先完成功能再提高精度。
4. 积累知识，总结规则。

例如：在调试到定点函数时，如机器人走波浪线、超调过大说明角度误差作用大于距离误差的作用，可适当减少其比例系数；反之如果机器人走大弧线（绕圈）说明对角度误差的调节能力太小。

周云龙、胡英重新整理

2002 年 1 月 22 日